AGNI: A framework for Distributed Scripting

M.Ranganathan

Multimedia and Digital Video Group

National Institute of Standards and Technology

AGNI

The AGNI Team

- M.Ranganathan
- Laurent Andrey
- Virginie Schaal
- External Collaborators:
  - Anurag Acharya UCSB
- Group Manager: J-P Favreau

AGNI

Goals of the AGNI Project

- Design and build a secure , extensible, fault-tolerant infra-structure for peer-to-peer event-driven (a-synchronous) applications.
- Develop peer-to-peer event-driven applications based on developed infra-structure.

Application Scenarios

- Distributed control.
- Conferencing, conference control.
- Distributed testing, logging and monitoring.
- Distributed Interactive Simulation.
- Network Management.

AGNI

Design Philosophy

- Distributed systems designer should be able to:
  - 1. Decide logical application structure.
  - 2. Application functionality.
- Independent of how application components are mapped to physical resources (separation of *logical design* and *physical design*).

AGNI

Design Considerations

- Extensibility and flexibility.
- Fault tolerance.
- Re-configurability.
- Security.
- Heterogeneity.

Event-oriented programming model.

- One-way message oriented.
- Message arrival triggers execution of event handler
  - Event handler can send message to another communication end-point.

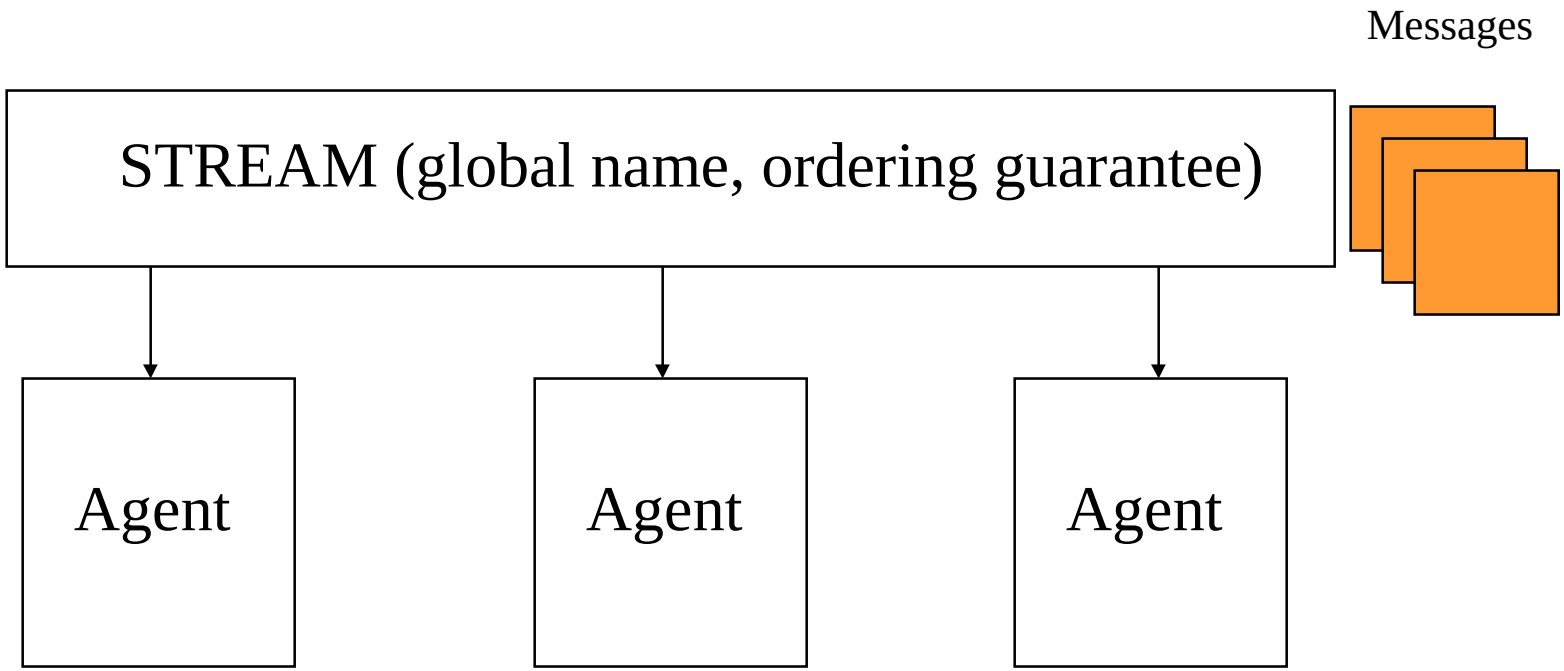
AGNI

Abstractions

- locations, streams, agents and events.
- Location maps to a machine.
- A Stream is a named communication end-point.
- An Event is a change in system state - potentially triggers agent execution.

AGNI

Distributed Streams



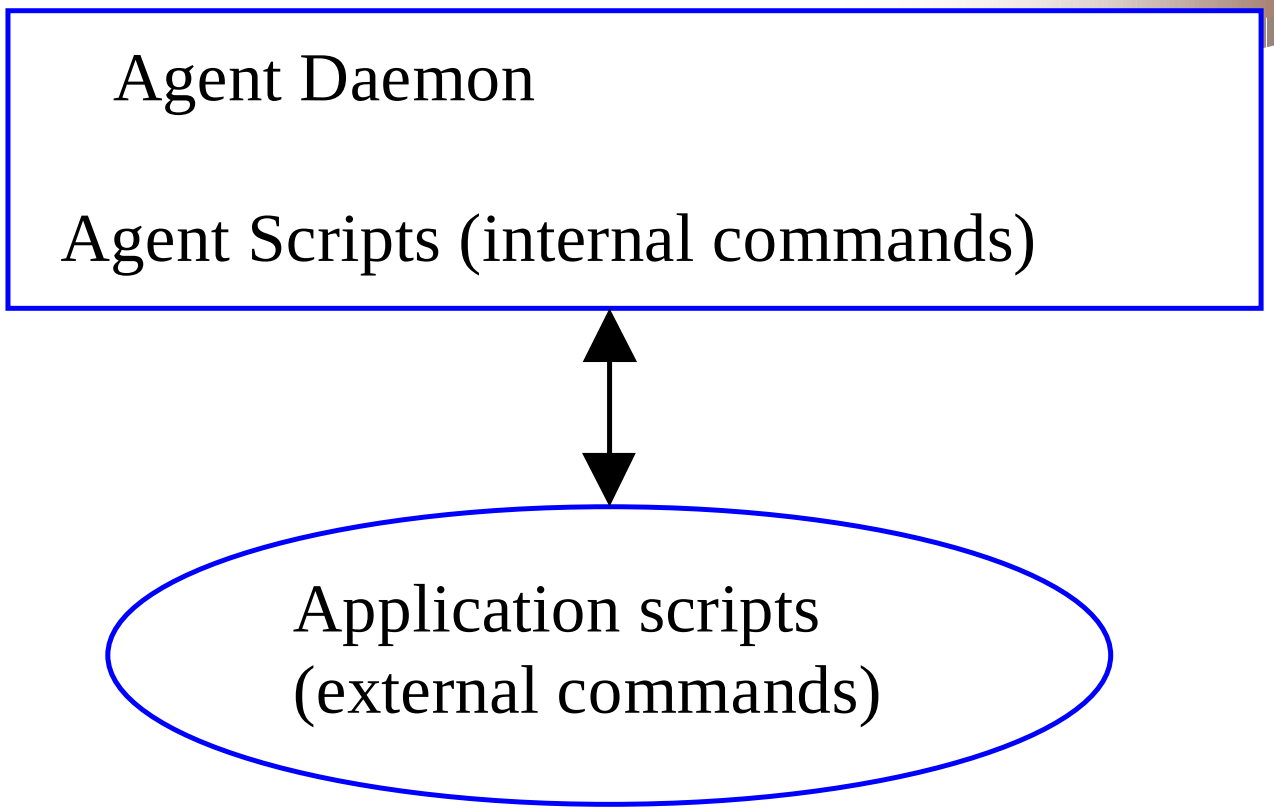
AGNI

Attributes of a Stream

- Ordering guarantee.
- Unique global name.
- Assigned Home Location but can live on any workstation (that allows it).
- Can migrate between workstations.
- Has 0 or more Agents associated with it.

AGNI

External and Agent Scripts



AGNI

Re-configurability

- Change the location of computational components dynamically while the system is in execution.
- Message ordering guarantees are maintained while system is being re-configured.
- Useful for latency reduction/load balancing.

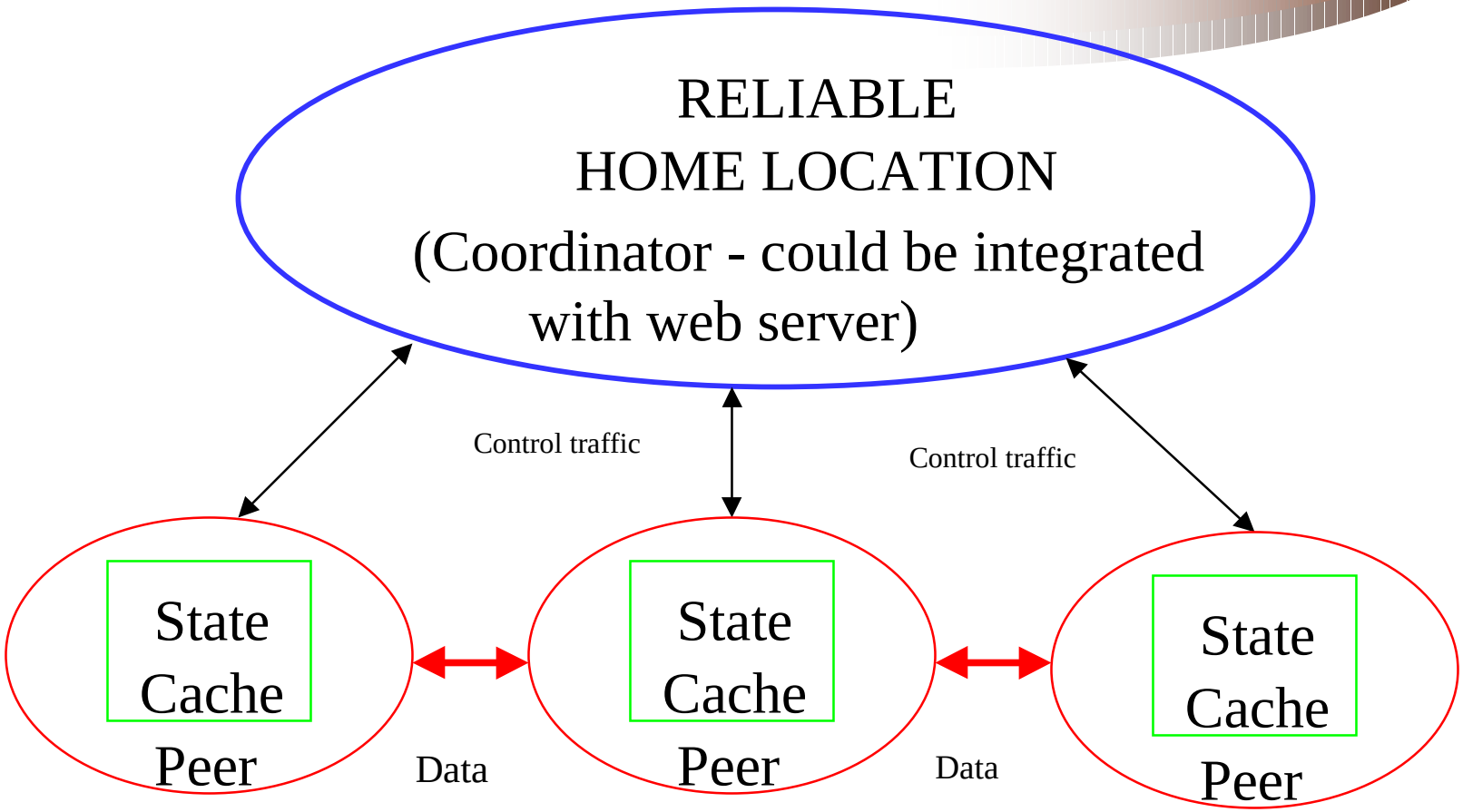
AGNI

System Features

- Agents may send messages to streams.
- External programs may send messages to streams.
- System can re-configure itself between messages.
- Multiple points of control are allowed.

AGNI

System Organization



AGNI

Agent script

- Agent script consists up to 5 TCL scripts:
  - on-init : Initializer.
  - on-append: Runs message is appended.
  - on-relocation: Runs at destination after move
  - on-failure: Runs at home node on failure.
  - on-exit: Finalizer (cleanup script).

AGNI

Resource-control Architecture

- Two-tiered Resource-control Architecture.
- Per daemon resource-controller:
  - Controls resource usage on a per Agent-Daemon basis.
- Per stream resource-controller:
  - Controls resource usage on a per-stream basis.

AGNI

Per-Daemon Resource Controller

- Stationary resource-control agent at each location
  - specified at startup time.
  - Can only be registered locally.
- Approves/denies stream creation/ arrival at a location.

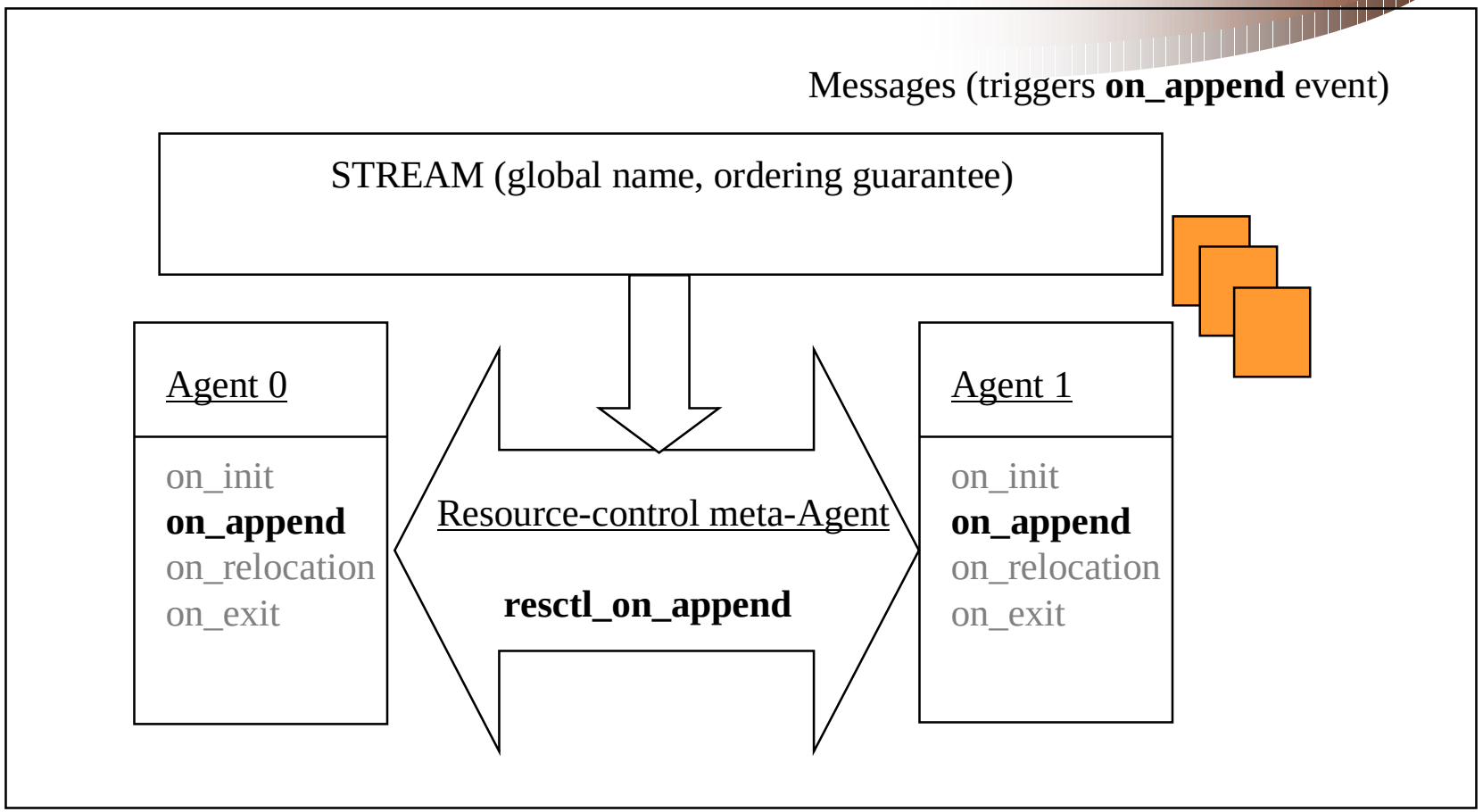
AGNI

Per-Stream Resource Controller

- Can be specified at Stream creation.
- Intervenes on append, agent attach, re-location, arrival.
- Works as a meta-agent.
  - Gets control before user-registered agents get to run.
  - Can decide which user-registered agents get to run.
  - Can execute commands in the context of the user agents.

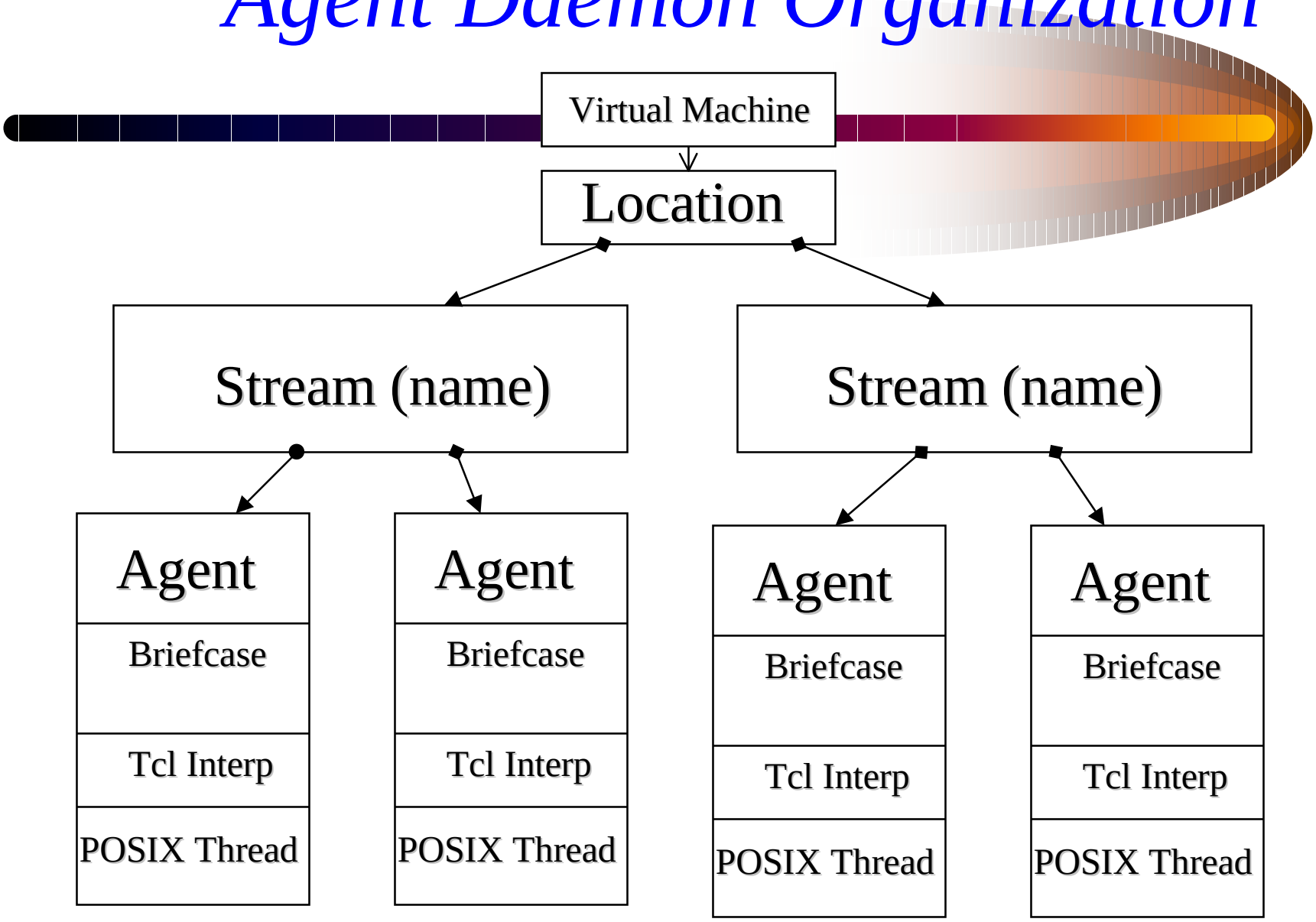
AGNI

Per-Stream Resource-Controller



AGNI

Agent Daemon Organization

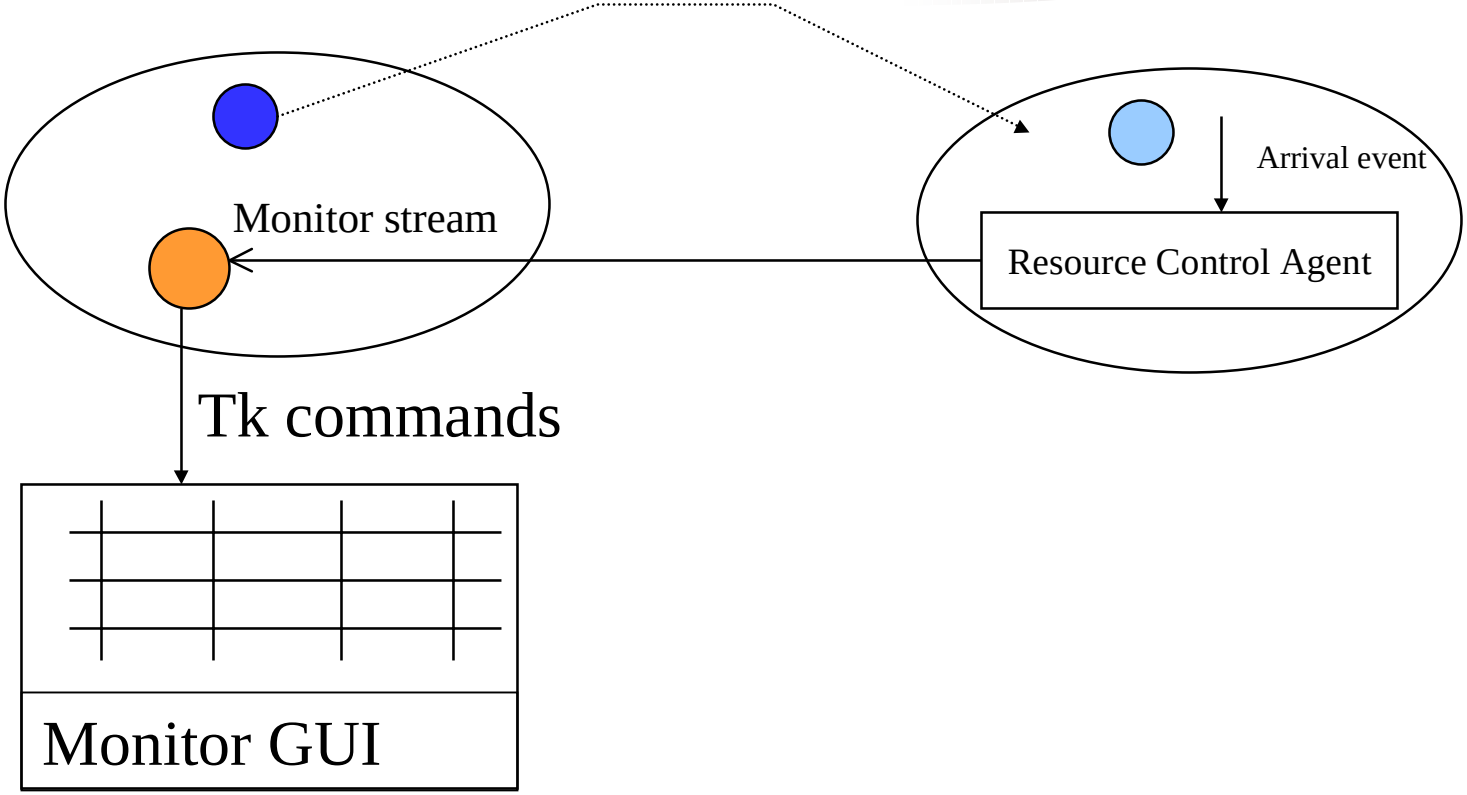


AGNI

Application : Agent Monitor

- Track the location of agents in the distributed system.
- Reactive GUI application.
- Allow user to easily deploy agents using a GUI driven management tool.

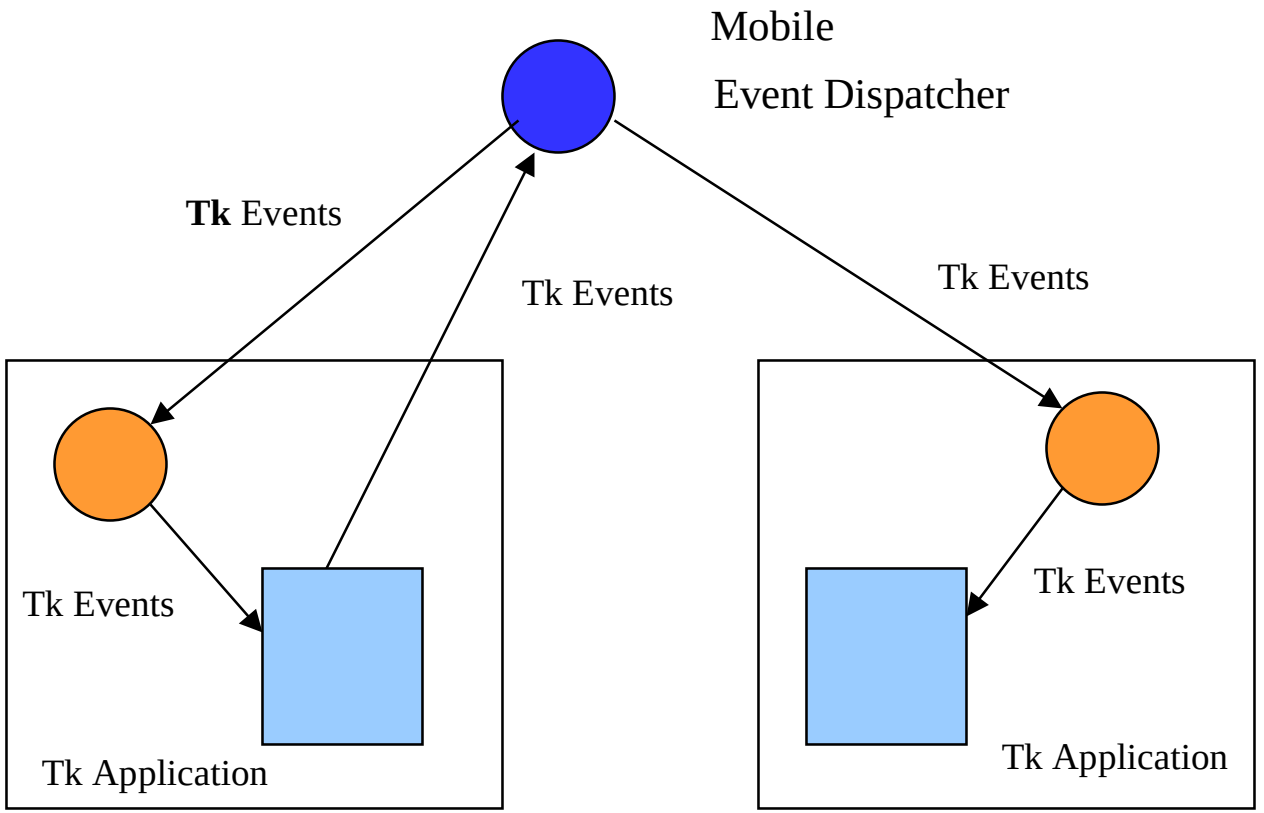
Agent monitor organization



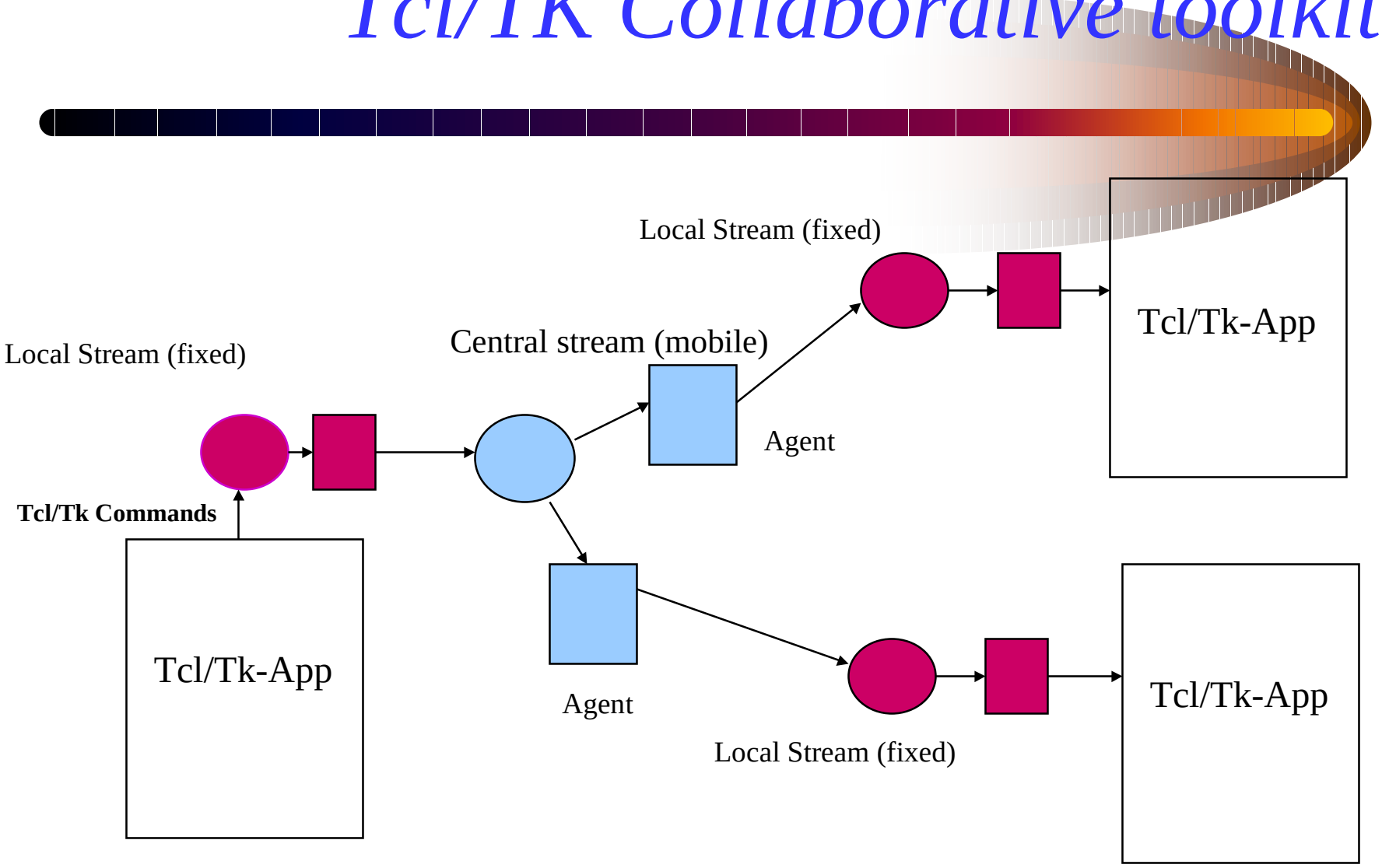
Application: Tk-Collaborative Toolkit

- Enables collaborative sharing of arbitrary TK applications.
- Self-reconfiguring distributed application.
- Reconfigures itself to minimize latency for the interactive user.

Application: Tk-Collaborative Toolkit



Application: Tcl/TK Collaborative toolkit



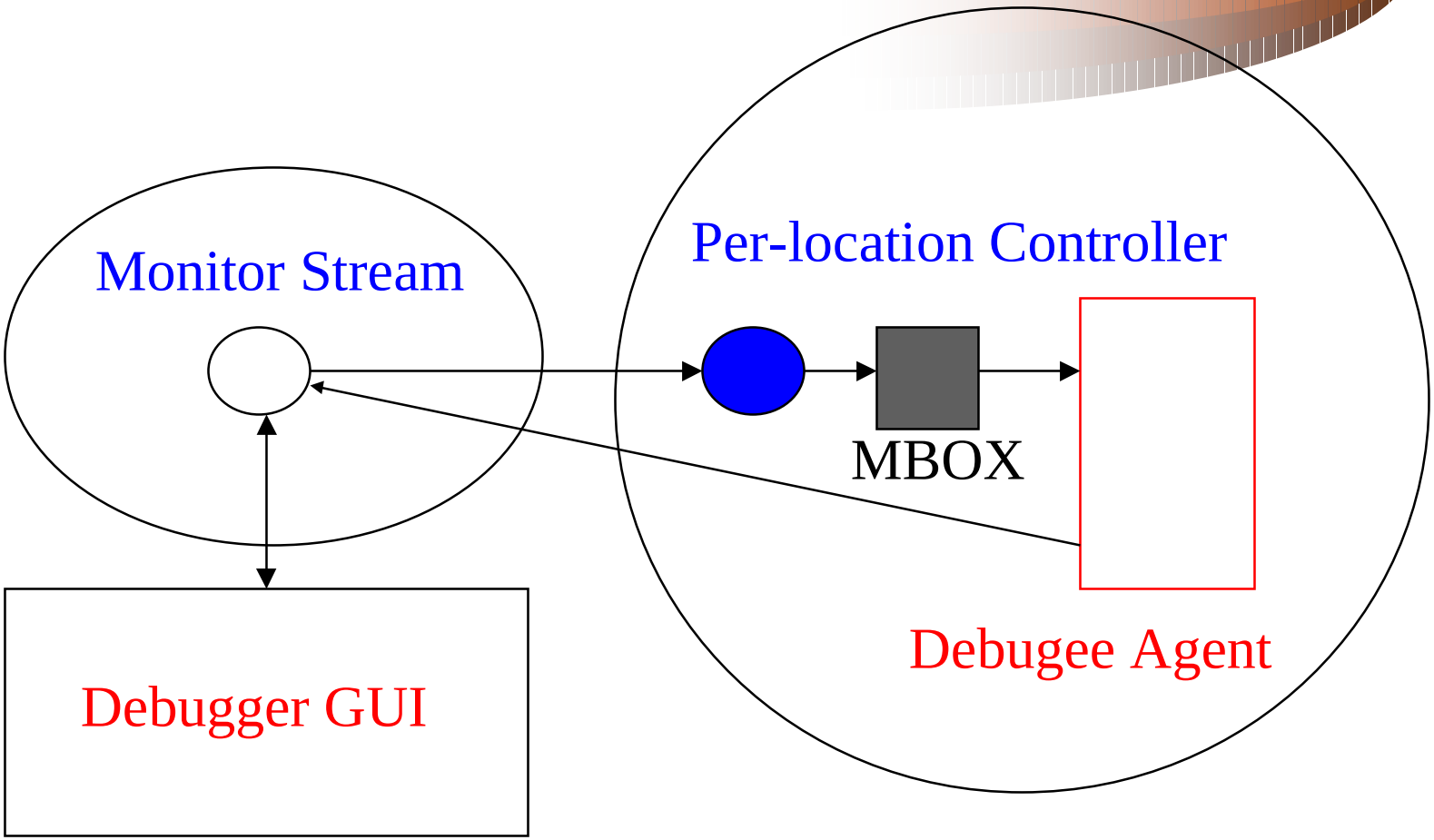
AGNI

Application: Agent Debugger

- A tool to debug Agent Scripts.
- Extension of tcl-debug debugger.
- Goals:
  - Location transparency.
  - Global stepping.
  - Global conditional breaks.

AGNI

Agent Debugger Organization



AGNI

Current Status.

- Initial prototype of agent system with limited features.
  - Lacks fault tolerance
  - Uses tcp for message passing (lacks scalability).
  - Limited resource-control model.
  - Lacks security.
- A few applications have been developed.

Future Work Includes

- Application-driven extension of system features.
  - Adaptive applications for networks.
  - Virtual Microscope.
  - Distributed testing of collaborative systems.
- Improving the communication scalability.
- Fault-tolerance.
- Security.